Computational phonology

A constraint-based approach

by

STEVEN BIRD

University of Edinburgh

CAMBRIDGE
UNIVERSITY PRESS

Published by the Press Syndicate of the University of Cambridge
The Pitt Building, Trumpington Street, Cambridge CB2 1RP
40 West 20th Street, New York, NY 10011-4211, USA
10 Stamford Road, Oakleigh, Victoria 3166, Australia

First published 1995

Printed in Great Britain at the University Press, Cambridge

A catalogue record for this book is available from the British Library

Library of Congress cataloguing in publication data

Bird, Steven.
Computational phonology: a constraint-based approach / Steven Bird
p. cm. – (Studies in natural language processing)
Includes bibliographical references and indexes.
ISBN 0 521 47496 5 (hardback)
1. Grammar, Comparative and general – Phonology – Data processing.
2. Computational linguistics. 3. Constraints (Artificial intelligence)
4. Language and logic. I. Title. II. Series.
P217.3.B57 1995
414'.0285–dc20 94-30865 CIP

ISBN 0 521 47496 5 hardback

SGB

Contents

1 Introduction

The last two decades have witnessed a vigorous growth of new descriptive notational devices in phonology. These devices have had enormous heuristic value in helping practitioners to see and intuitively understand complex phenomena. However, linguistic notations should be 'perfectly explicit' and ought not 'rely on the intelligence of the understanding reader' (Chomsky, 1965, 4). It is not clear that modern non-linear phonology, to any great extent, meets these fundamental requirements of generative grammar. If current work in computational phonology and speech technology is focused on the outdated SPE model (Chomsky and Halle, 1968) it is because nothing more recent has surpassed SPE's formal explicitness. Therefore, it is time for these new phonological frameworks to be placed on a formal footing.

Those who are suspicious of formalism may cry foul at this point. After all, many a good linguistic insight has been buried under a barrage of definitions and theorems, and a preoccupation with technical hygiene may blinker one's vision of what is really going on. However, the solution is not to retreat to a position where *formulating* a description is synonymous with *formalising* a description. Rather, we need to recognise that formalisation has considerable heuristic value of its own. After all, linguistic theorising in this century has been characterised – possibly even driven – by a tension between attempts at rigorous theories of linguistic structure and attempts to formulate intuitively sensible descriptions of linguistic phenomena.

Beyond this, formalisation is fundamental to the empirical basis of the field. The widespread practice of testing an empirical generalisation on isolated examples leads to unstable theories which are restricted to small fragments of a language. If, as noted with regret by Anderson (1989, 803), outside observers do not always take phonology seriously, then an important reason is different notions of what a scientific theory is and does. As we shall see below, a phonological theory which can be implemented on a computer can meet the dual requirements of rigour and non-trivial empirical content which much current work has unsuccessfully striven to achieve.

Underlying these concerns is the goal of constructing grammars which do not favour generation at the expense of recognition, or vice versa. This connects with the familiar debate about the metatheoretical undesirability of extrinsically ordered rules (Koutsoudas, Sanders and Noll, 1974; Hooper, 1976), and with earlier complaints that the derivational stance of generative phonology was inherently process-oriented. Despite claims to the contrary, many current phonological theories remain performance models. They enumerate the steps which must be taken in moving from a lexical form to a surface form, borrowing heavily on the now dated flowchart model of computation. Crucially, there is no guarantee that such rule systems work in reverse.[1] If we accept that linguistics is the study of that knowledge which is independent of processing tasks, then the statements of a linguistic theory ought to have a declarative semantics: an interpretation which is expressed solely in terms of the utterances which are licensed by theory. Of course, if a theory is going to be useful its statements should also have one or more procedural interpretations, but these ought not to be mistaken for the linguistic theory itself.

Readers with a background in computational syntax and semantics will be wondering how this computational phonology could fit into an overarching computational grammar framework. Here, our starting point is provided by the work of Deirdre Wheeler and Emmon Bach,[2] who showed how the principles of Montague grammar can be applied to phonology. However, the aim here is not to perform this integration of phonology and grammar, but rather to do phonology in such a way that this integration is possible. Therefore, a monostratal approach[3] to phonological description

[1] This non-reversibility is a general result, which Bear (1990) has demonstrated for Klamath (Halle and Clements, 1983, 113).

[2] Wheeler (1981, 1988); Bach and Wheeler (1981); Bach (1983).

[3] See §1.4.5 for an explanation of the term 'monostratal'.

has been adopted since this is a requirement for a phonological framework to be integrated into existing constraint-based grammar frameworks.[4]

In this connection it is necessary to introduce two distinctions. The first is the DESCRIPTION/OBJECT distinction: an expression of a linguistic theory DENOTES the class of utterance tokens which SATISFY that expression. These expressions are combined using familiar logical connectives. While there is a fundamental difference in kind between descriptions and objects, and so one might imagine that this configuration is actually polystratal, there remains only a single level of linguistic description. This state of affairs contrasts with the procedural model of traditional generative phonology in which there is no principled upper bound on the number of intermediate levels of description. Frameworks which build in this distinction are sometimes called CONSTRAINT-BASED because their linguistic descriptions act in concert to mutually constrain the solution space.

A second distinction is that of FRAMEWORK versus THEORY. A linguistic framework is essentially a formal notation in which linguistic theories can be stated. As such, a framework makes no empirical claims of its own, though a good framework should facilitate the expression and evaluation of such claims. Just as two theories which make contradictory claims can be expressed in the same framework, a given theory can potentially be encoded in a variety of different frameworks. A computational benefit of frameworks is that once a framework is implemented, a whole family of theories can be easily expressed within it, and it is not necessary to write whole implementations from scratch for each new theory.

These are the essential ingredients of what I shall call CONSTRAINT-BASED PHONOLOGY, a term derived from the established fields of constraint-based grammar and constraint logic programming. It is hoped that the eventual payoff of work in this vein will be the construction of rigorous and empirical phonological theories along with the construction of integrated systems for speech and language processing. However, in the light of this aspiration the immediate goals are more humble. After providing the necessary background material in chapter 1, a logical foundation for phonology cast in a language of classical first-order predicate logic is presented along with

[4]For example, Generalised Phrase Structure Grammar (Gazdar et al., 1985), Categorial Unification Grammar (Uszkoreit, 1986), Head-Driven Phrase Structure Grammar (Pollard and Sag, 1987), and Unification Categorial Grammar (Calder, Klein and Zeevat, 1988). Some evidence of initial progress in this direction can be found in Bird (1992); Bird and Klein (1994).

a model-theoretic semantics (chapter 2). This constitutes the framework in which subsequent theorising is couched. In chapter 3 this is applied to a sampler of frequently cited phenomena which might be thought to present obstacles to a monostratal approach. Chapter 4 consists of an investigation of recent proposals in the theory of feature geometry, followed by the formalising and interfacing of Sagey (1986) and Browman and Goldstein (1989). These two chapters exemplify the framework by showing how theories can be expressed. The implementation of the model used throughout this work is presented in chapter 5, in the form of a temporal constraint-solver.

The remainder of chapter 1 is structured as follows. First, an intuitive introduction to generative phonology is presented in §1.1, focusing on the progression from linear to non-linear phonology (the autosegmental version). In §1.2 the formal adequacy of the autosegmental notation is discussed and found to be deficient. This deficiency is a stumbling block for any attempt to provide a computational interpretation of the autosegmental framework. Section 1.3 presents an overview of current work in computational phonology and §1.4 is a discussion of constraint-based phonology. The history of constraints in phonology is surveyed in §1.5.

1.1 Introduction to phonology

Segmental phonology derives its name from the fundamental hypothesis that – for linguistic purposes – speech can be sliced into a linear sequence of events, like beads on a string, known as SEGMENTS. Although we will have good reason to challenge this hypothesis later, its initial appeal comes from the existence of alphabetic writing systems, which involve more or less discrete tokens arranged in sequence.

The first illustration of segmental phonology will be based on the formation of regular plural in English. The following table shows various English plural nouns, and alongside each noun is the letter *s* or *z*, indicating how the plural is actually pronounced.

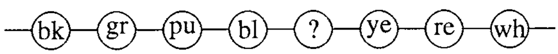
word	suffix	word	suffix
caps	s	cabs	z
mats	s	pads	z
backs	s	bags	z

Observe that for words ending in *p*, *t* and *k* the plural is *s*, while for words ending in *b*, *d* and *g* the plural is *z*. In fact, this observation holds for most English nouns ending in these consonants. The question is, how do speakers know when to use *s* and when to use *z*? The answer given by traditional segmental phonology is to provide the following rule:

(1.1) $\quad s \rightarrow z \,/\, \{b, d, g\}\_\_\_$

This means that an *s* becomes a *z* in the context '{b, d, g}___'. To understand the context, imagine the *s* being in the position of the underline. In other words, an *s* is only affected if it follows a *b*, *d* or *g*. Rules like (1.1) have a transformational character, and generative phonology has sometimes been referred to, somewhat pejoratively, as 'transformational phonetics' (Foley, 1977). Rule (1.1) is not an entirely satisfactory explanation of the phenomenon. Could we arbitrarily modify some details of this rule and get an equally plausible rule? The answer is no.

To see how to proceed from here we must return to the beads-on-a-string analogy. Imagine we have a colourful necklace, as shown below. Colours are denoted by two-letter abbreviations. What is the colour of the fifth bead?



The sequence looks completely random, yet if we decompose the colours into their primary components, as shown below, it is possible to guess the pattern and discover that the fifth bead must be orange.

(1.2)

	bk	gr	pu	bl	or	ye	re	wh
R	+	−	+	−	+	−	+	−
Y	+	+	−	−	+	+	−	−
B	+	+	+	+	−	−	−	−

This technique of resolving colours into their components also works well in the discovery of patterns in sequences of phonological segments.

Let us return to the English plural. Here we have a collection of segments, like *s* and *b*, but how should they be decomposed? Consider *s* and *z*. They are identical except for the property of VOICING. The segment *z* involves a periodic vibration of the glottis, while *s* does not. English has several pairs of voiceless/voiced consonants, as shown in the following table:[5]

	lips teeth	tongue tip	tongue body palate
plosives	p/b	t/d	k/g
fricatives	f/v	s/z	ʃ/ʒ

So words ending in voiceless plosives take the plural form *s*, which is the voiceless member of the {*s*, *z*} pair, while words ending in voiced plosives take the plural form *z*, the voiced member of {*s*, *z*}. We call this an ALTERNATION involving *s* and *z*, and write *s*∼*z*, '*s* alternates with *z*'. Here, we can observe that the consonant cluster must *agree* in voicing; *i.e.* a voicing ASSIMILATION has taken place. In segmental phonology, this observation can be expressed as the rule in (1.3) below.

(1.3) $\mathrm{s} \rightarrow$ [+voice] / [+voice] ___

This still does not quite capture the idea of assimilation, since we could just as easily have had a version of the above rule where the context was [–voice], which would be implausible. So instead the following notation is used, employing a Greek variable α which ranges over the values $+$ and $-$, and a symbol *S* to represent a segment like *s* and *z* which is unspecified for voicing.

(1.4) $\mathrm{S} \rightarrow [\alpha\mathrm{voice}]$ / $[\alpha\mathrm{voice}]$ ___

For the next innovation, it is necessary to return to the coloured beads. Consider again the strand of beads we saw above, repeated as (1.5a). Suppose we were to view these beads through tinted glasses. The strand in (1.5b) corresponds to yellow glasses, while that in (1.5c) corresponds to red glasses.

[5]The ʃ and ʒ consonants are the middle consonants of the English words *machine* and *regime* respectively.

(1.5)
a. —bk—gr—pu—bl—or—ye—re—wh—
b. —bk—gr—bk—gr—or—ye—or—ye—
c. —bk—bk—pu—pu—or—or—re—re—

Observe that the black bead in (1.5a) remains black in (1.5b,c) regardless of which glasses are used, while the other beads are affected in different ways. For example, the blue bead in (1.5a) appears green in (1.5b) and purple in (1.5c).

Returning to phonology, let us consider the vowels of Turkish. These vowels are tabulated below,[6] along with a decomposition into DISTINCTIVE FEATURES. Notice that this table is the same as the colour chart in (1.2) but with different labels. We have already met the distinctive feature [voice]. Here we meet the features [high], [back] and [round]. The features [high] and [back] relate to the position of the tongue body in the oral cavity. The feature [round] relates to the rounding of the lips.

(1.6)

	u	o	ü	ö	ı	a	i	e
high	+	−	+	−	+	−	+	−
back	+	+	−	−	+	+	−	−
round	+	+	+	+	−	−	−	−

The analogy with beads and tinted glasses will help us to understand the vowel patterns of Turkish words. Think of each word as a strand of beads, but where each word also supplies a different pair of tinted glasses to use. Now, look at the following Turkish words. Pay particular attention to the four versions of the possessive suffix.

ip	*rope*	ipin	*rope's*
kız	*girl*	kızın	*girl's*
yüz	*face*	yüzün	*face's*
pul	*stamp*	pulun	*stamp's*

The suffix has the forms *in*, *ın*, *ün* and *un*. Observe that the vowel of the suffix agrees with the vowel of the stem, and observe that the consonants do not change (cf. the black beads). Now although the possessive nouns (in the third column) have two vowels, it is appealing to imagine that there

[6] Note that there is a distinction made in the Turkish alphabet between the dotted *i* and the dotless *ı*. This *ı* is a high, back, unrounded vowel that does not occur in English.

is just the one vowel which 'colours' the whole word, just as coloured glasses tint an entire view. As we have seen, Turkish actually has eight vowels, but only *four* forms of the possessive suffix. Thus, the suffix vowel is not always identical to the stem vowel, as the above data may lead one to suspect. As an example of the situation where the suffix vowel differs from the stem vowel, consider the word *el* 'hand' which has the possessive form *elin*, rather than **elen*. With the help of table (1.6) the reader should be able to determine which forms of the suffix are used for the words *çan* 'bell', *köy* 'village' and *son* 'end'. Consult the footnote for the solution.[7]

Let us see how segmental phonology might express this vowel patterning. We begin by supposing that the vowel of the possessive affix is only specified as [+high] and not specified for the features [back] and [round]. *C* denotes any consonant.

$$\text{(1.7)} \quad [\text{+high}] \longrightarrow \begin{bmatrix} \alpha\text{back} \\ \beta\text{round} \end{bmatrix} / \begin{bmatrix} \alpha\text{back} \\ \beta\text{round} \end{bmatrix} C\_\_$$

So long as the stem vowel is specified for the properties [high] and [back], this rule will make sure that they are copied on to the affix vowel. However, this approach misses out on the idea that there is just one property that is spread over the whole word. It also allows nonsense rules where the Greek variables are switched around:

$$[\text{+high}] \longrightarrow \begin{bmatrix} \alpha\text{back} \\ \beta\text{round} \end{bmatrix} / \begin{bmatrix} \beta\text{back} \\ \alpha\text{round} \end{bmatrix} C\_\_$$

Another approach available to segmental phonology, which avoids the use of Greek variables, is to suppose that the DEFAULT values for [back] and [round] are [–back] and [–round] respectively. Then we can have independent rules for [round] and [back] and these only refer to the non-default – or MARKED – values:

$$[\text{+high}] \longrightarrow [\text{+round}] / [\text{+round}]\, C\_\_$$

$$[\text{+high}] \longrightarrow [\text{+high}] / [\text{+high}]\, C\_\_$$

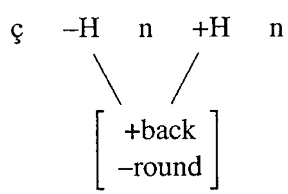
[7]

el	*hand*	elin	*hand's*
çan	*bell*	çanın	*bell's*
köy	*village*	köyün	*village's*
son	*end*	sonun	*end's*

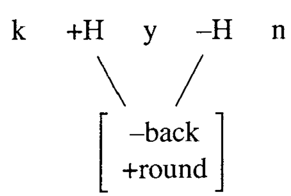
Here we are starting to see the breakdown of the fundamental hypothesis of segmental phonology, introduced at the beginning of this section, which states that speech can be segmented into a linear sequence of discrete events. Let us suppose for a moment that speech *cannot* be sliced up in this way. In other words, suppose that picking boundaries between segments based on a feature like [back] gives different results to picking boundaries based on [voiced]. If we cut the speech stream into slices we would expect to find that adjacent slices are not independent of each other but share many properties. It is plausible that we might be able to come up with rules to express the relationship between adjacent slices. Crucially though, the more rules we had to come up with, the less plausible would be our starting assumption that slicing up the speech stream in this way helps to understand what is going on. Now, segmental phonology is sometimes assumed to have been a success because of the elaborate systems of rules and derivations that it inspired. However, the existence of such a rich armoury of constraints, rules and processes is simply an epiphenomenon. Paradoxically, the 'success' of segmental phonology is actually evidence that the segmental hypothesis is untenable.

Let us now abandon segmental phonology and try to give some content to our intuition that Turkish words are coloured with certain properties like [+back] or [–round]. Consider the words *çanın* and *köyün*. We could depict them thus:

(1.8) a.



b.

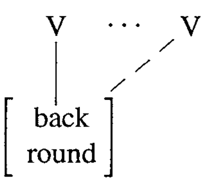


Entities like [+back,–round] that function over extended regions are known as PROSODIES, and this kind of picture is called a NON-LINEAR representation.

The notion of prosodies dates back to the mid-century work of Harris (1944), Pike and Pike (1947), Firth (1948) and Hockett (1954). Today this notion is present in a wide range of phonological frameworks.[8] Here we shall see the fundamentals of just one of these models, known as AUTOSEGMENTAL PHONOLOGY, so called because it views the prosodies as autonomous segments, or simply AUTOSEGMENTS.

In autosegmental phonology, diagrams like those we saw above are known as CHARTS. A chart consists of two TIERS, along with some ASSOCIATION LINES drawn between the autosegments on those tiers. The NO-CROSSING CONSTRAINT is a stipulation that association lines are not allowed to cross.[9] AUTOSEGMENTAL RULES are procedures for converting one diagram into another, by adding or removing association lines and autosegments. A rule for Turkish vowel harmony might look like the following, where *V* denotes any vowel:

(1.9)



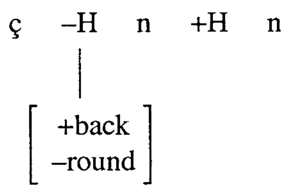
Rule (1.9) applies to any diagram containing (1.10a) and produces a similar diagram with an additional line (1.10b), which is the same as (1.8a).

---

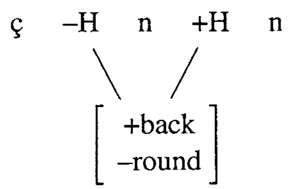
[8] Dynamic Phonology (Griffin, 1985), Dependency Phonology (Anderson and Durand, 1987), Autosegmental and Metrical Phonology (Goldsmith, 1990) and Government Phonology (Kaye, Lowenstamm and Vergnaud, 1990).

[9] As we shall see in §2.3.3, this constraint is a consequence of the temporal nature of autosegmental diagrams.

(1.10) a.

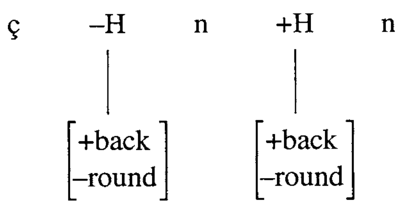


b.



Notice that there is nothing to stop us from constructing an equivalent representation for *çanın* in (1.11) which gets us back to segmental phonology.

(1.11)



In fact there is a constraint, called the OBLIGATORY CONTOUR PRINCIPLE (Leben, 1973), which rules out such diagrams. It requires that two autosegments must be distinct if they are adjacent on the same tier.[10]

This concludes the introduction to segmental and autosegmental phonology. Now we shall move on to look at some technical problems with autosegmental phonology.

[10] See Yip (1988) for a survey of the obligatory contour principle.

1.2 The formal adequacy of autosegmental notation

Autosegmental phonology is fundamentally a collection of descriptive notational devices, along with conventions about their application and interpretation. Goldsmith states:

> It would not be wrong, in fact, to summarise the entire goal of autosegmental analysis as being the reduction of natural phonological processes to changes that can be expressed in the minimal autosegmental notation... (Goldsmith, 1990, 73f)

Given the central status of notation in autosegmental theory, the imprecise definition and widespread abuse of notation in the literature is somewhat disconcerting. The reader seeking a *precise* understanding of the basic elements of autosegmental representations and rules will often be left floundering. Is the association relation reflexive, symmetric or transitive?[11] What does the no-crossing constraint really mean? What does the *absence* of an association line mean? Are diagrams to be thought of as descriptions or as objects?

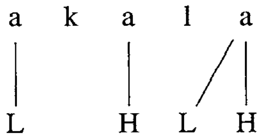
Here we shall consider just the no-crossing constraint. One interpretation is that the no-crossing constraint blocks the application of a rule which would result in crossing lines (Goldsmith, 1990, 30). Another view is that the no-crossing constraint *repairs* ill-formed representations by causing the deletion of the line crossed by the new line (Goldsmith, 1990, 47,79).[12] For example, consider the effect of a rule like (1.9), which spreads the initial tone of a word to subsequent vowels, as applied to (1.12).

---

[11] These properties are explained in §2.3.3.

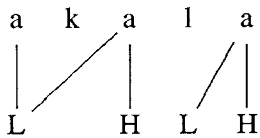
[12] This ambiguity between blocking versus repair views has arisen in other areas of phonology, most notably in connection with the obligatory contour principle, which requires that two adjacent tier elements must not be identical (Goldsmith, 1976, 36). Leben's (1973) original definition of the obligatory contour principle clearly views it as a repair strategy which alters a representation containing adjacent identical elements by fusing them into one. However, McCarthy (1986, 222) has advocated the opposite view, where the OCP simply blocks a derivation. Yip (1988), on the other hand, advocates both views.

(1.12)



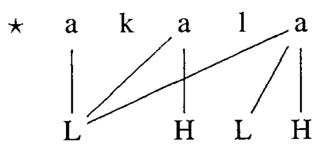
On the first application, a line from the *L* to the second *a* is added:

(1.13)

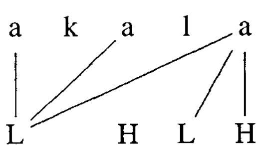


Under the 'blocking' view of the no-crossing constraint, the rule could not re-apply, since linking the *L* to the third *a* would cause a line crossing. Under the 'repair' view, there are two further steps. In (1.14a) a line which violates the no-crossing constraint is added (the asterisk indicates this ill-formedness), and in (1.14b), the old line is automatically deleted leaving the first *H* tone unassociated.

(1.14) a.



b.



This ambiguity between the blocking and repair interpretations arises because there is only an informal connection between the well-formedness condition and a derivation it is understood to license. Therefore, the question of interpreting the no-crossing constraint needs to be addressed (see §2.3.3). Since derivations are understood to be deterministic, both interpretations cannot co-exist.

This concludes our discussion of the formal adequacy of the autosegmental notation. There are several other technical problems with current notational practice in autosegmental phonology; Bird and Ladd (1991) give a detailed critique. Once these problems are solved, the way is open for the computational implementation of autosegmental phonology. In the next section an overview of current developments in computational phonology is presented.

1.3 Computational phonology

1.3.1 Motivation

The practising phonologist is frequently beset by two problems regarding data and analysis. First, a realistically sized corpus of data is difficult to maintain and access if it is stored on paper. Putting such a corpus online is a solution, but some difficult technical problems need to be solved first.[13] Second, a realistically sized analysis is virtually impossible to check by hand. The widespread practice of testing a few interesting cases is unreliable and is no substitute for an exhaustive check. Here again, automation promises to provide a solution, provided that phonological analyses can be represented on computer. These considerations motivated some of the earliest work on computational phonology, such as Bobrow and Fraser's *Phonological Rule Tester*:[14]

> Although the theory [of SPE] has been designed and modified to enable the linguist to state generalizations about a language in a simple and revealing way, an account of some significant portion of a language often results in a complex and interdependent set of rules. Consequently, it becomes

[13] Griswold (1992) describes the following classic problems of multilingual computing: character rendering (character shapes, context sensitivity, diacritic placement), line rendering, data entry, internal storage, alphabetising, searching, word demarcation.

[14] Other early SPE implementations are Friedman and Morin (1971); Roosen-Runge and Kaye (1973).

> more difficult for the linguist to evaluate the work he has done. In fact, linguists have reached the point today where the detail of analysis makes it impracticable to evaluate by hand even a small set of rules. In this paper the design and implementation of a system to alleviate the problem of rule evaluation for the linguist in the area of phonology are presented... (Bobrow and Fraser, 1968, 766)

Another source of motivation for computational phonology comes from the field of speech technology. If speech recognition systems incorporate a rule-based model, it is almost invariably the SPE model (e.g. Allen et al., 1987). Some have argued that this approach has foundered and that speech technologists should look to phonology for new conceptual machinery (Kaye, 1989; Kornai, 1991). In a similar way, the field of natural language processing is currently limited to a small group of languages which lack morphophonological alternations in the orthography. Gazdar made a similar observation:

> Until recently there has been relatively little computational linguistic work on morphology or phonology, in contrast to the large amount of work on syntax and speech. The explanation for this state of affairs is not hard to find: most computational linguistics has dealt with English, and the latter's inflectional morphology is impoverished. Consequently, the English lexicon can be treated, for many practical purposes, simply as a list with the spelling and phonetics already encoded. Derivational morphology can be ignored ('if it occurs, list it!'), and inflectional morphology can be reduced to a few ad hoc devices to realise -S and -ED. This kind of strategy collapses completely, of course, when confronted with a language like Finnish or Turkish. (Gazdar, 1985, 604)

This is a challenge to develop a computational phonology which can be applied to the full diversity of the world's languages. Once this is done, the achievements of the field of natural language processing will have far greater applicability.

A final area of motivation comes from the long-term prospect of having integrated speech and language systems. Phonology is a potential link between the speech technology community and the natural language processing community. However, as we have seen, contemporary phonology is inadequately formalised to play this mediating role. Perhaps computational phonology will ultimately bridge the gap between these two independent areas of technological development.

In the rest of this section, the recent developments in computational phonology are surveyed.[15] We begin by considering finite-state methods, one of the earliest approaches.

1.3.2 Finite-state methods

The idea of employing FINITE-STATE TRANSDUCERS (FSTs) to represent the rule systems of generative phonology was proposed in the early 1980s by Kaplan and Kay in unpublished work[16] (though now published as Kaplan and Kay 1994), which connected with Johnson's demonstration that the relations described by generative phonological rules are REGULAR (Johnson, 1972). Koskenniemi (1983a, 1983b, 1984) proposed an FST model where rules could refer to *both* surface and lexical contexts, but that these were the only levels of representation, and so there could be no feeding or bleeding relationships between rules.[17] Koskenniemi also proposed a high-level notation for rules which could be compiled into transducer specifications (Koskenniemi, 1985). Antworth (1990) gives a detailed exposition of the rule notation, the transducer specifications and the compilation process. Ritchie et al. (1992) and Sproat (1992) also give expositions of the two-level model, while Pulman and Hepple (1993) present a two-level system incorporating a unification-based representation of segments. Here we shall see the rule notation and its application to Turkish vowel harmony.

The rule notation employs pairs of symbols, consisting of a LEXICAL symbol and a SURFACE symbol. Rules express constraints on the distribution of these pairs. There are three types of rule, as shown in (1.15).

[15] See Bird (1994b) for a differently structured survey.

[16] See Kay (1983, 100–4).

[17] Two rules r_1 and r_2 are in a FEEDING (resp. BLEEDING) relationship if the application of r_1 creates (resp. destroys) the context necessary for r_2 to apply (Kiparsky, 1968, 196ff).